NLP Assignment 2
Language model implementation

In this assignment, you are asked to write a program to utilize the training data to generate your *n*-gram language model with Good-Turning Smoothing, and to use your *n*-gram language model to calculate the probability of a sentence.

Given a sentence $s = w_0 ... w_{n-1}$ with length *n*, we can use the training data to calculate the count of w_i (unigram) and $w_i w_{i+1}$ (bigram). And we can also calculate $P(w_i)$ and $P(w_{i+1} | w_i)$, which are defined as:

$$P(w_i) = \frac{Count(w_i)}{\sum_j Count(w_j)}$$

$$P(w_i, w_{i+1}) = \frac{Count(w_i, w_{i+1})}{\sum_j Count(w_j, w_{j+1})}$$

$$P(w_{i+1} \mid w_i) = \frac{P(w_i, w_{i+1})}{P(w_i)}$$

where $Count(w_i)$ and $Count(w_i, w_{i+1})$ is the count of w_i and $w_i w_{i+1}$ in the training data.

The probability of *s* is defined as:

$$P(s) = P(w_0) \times P(w_1 | w_0) \times P(w_2 | w_1) \times ... \times P(w_{n-1} | w_{n-2})$$

To avoid the probabilities of unseen unigrams or bigrams being zero, we use Good-Turning Smoothing to calculate the smooth count of w_i and (w_i, w_{i+1}), $Count^*(w_i)$ and $Count^*(w_i, w_{i+1})$.

$$Count^*(w_i) = (Count(w_i) + 1) \times \frac{N_{Count(w_i)+1}}{N_{Count(w_i)}}, \text{ if } Count(w_i) < k$$

$$Count^*(w_i) = Count(w_i), \text{ if } Count(w_i) \geq k$$

$$Count^*(w_i, w_{i+1}) = (Count(w_i, w_{i+1}) + 1) \times \frac{N_{Count(w_i, w_{i+1})+1}}{N_{Count(w_i, w_{i+1})}}, if\ Count(w_i, w_{i+1}) < k$$

$$Count^*(w_i, w_{i+1}) = Count(w_i, w_{i+1}), \qquad if\ Count(w_i, w_{i+1}) \geq k$$

where N_c is the frequency of frequency *c*, which means the number of words appearing *c* times.

Also, the $P(w_i)$ and $P(w_{i+1}, w_i)$ can be redefined as:

$$P(w_i) = \frac{N_1^u}{N_0^u N^u}, \text{ if } Count(w_i) = 0, \quad N_0^u = V - N_1^u - N_2^u - N_3^u \ldots$$

$$P(w_i) = \frac{Count^*(w_i)}{N^u}, \text{ if } 0 < Count(w_i) < k$$

$$P(w_i) = \frac{Count(w_i)}{N^u}, \text{ if } Count(w_i) \geq k$$

$$P(w_i, w_{i+1}) = \frac{N_1^b}{N_0^b N^b}, \text{ if } Count(w_i, w_{i+1}) = 0, \quad N_0^b = V^2 - N_1^b - N_2^b - N_3^b \ldots$$

$$P(w_i, w_{i+1}) = \frac{Count^*(w_i, w_{i+1})}{N^b}, \text{ if } 0 < Count(w_i, w_{i+1}) < k$$

$$P(w_i, w_{i+1}) = \frac{Count(w_i, w_{i+1})}{N^b}, \text{ if } Count(w_i, w_{i+1}) \geq k$$

where k is a constant (in this assignment, we set k=10), N^u is the count of unigram in training data, N^b is the count of bigram in training data, N_c^u is the is the frequency of frequency c in unigram, N_c^b is the is the frequency of frequency c in bigram, c, which means the number of words appearing c times, and V is the vocabulary size (we set V=80000).

The sample input/output is shown below.

```
Input sentence: IL-2 is a gene .
P*('IL-2 is a gene .') : 4.2651959979e-10
Input sentence: IL-2 is an gene .
P*('IL-2 is an gene .') : 3.97354252895e-16
```

Note: All punctuations are treated as words, so you should not remove them. Sentences are separated by newline, and sentences must be lowercase.

Finally, you need to provide a compressed file containing source codes, executable file, and introduction document which has to include the program environment. For your exe file, TA should be able to input some test string to evaluate your program.

Deadline: Oct. 16 at 11p.m.